

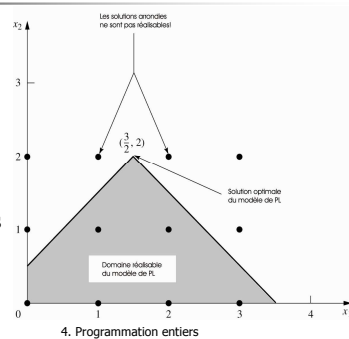
4. Programmation en nombres entiers  
 b. Séparation et évaluation progressive  
 c. Plans de coupes

Résolution de modèles entiers

- Programmation en nombres entiers = Programmation linéaire avec certaines variables à valeurs entières
- Pourquoi ne pas oublier les contraintes d'intégralité...
- Résoudre le modèle de PL ainsi obtenu (appelé *relaxation PL*)...
- Puis arrondir aux valeurs entières les plus près?
- Dans certains cas, ça peut fonctionner...
- Mais dans d'autres cas, cette *méthode par arrondissement* est désastreuse!

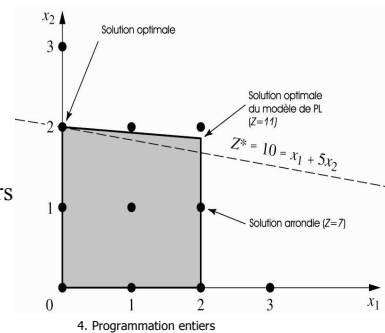
Arrondissement : exemple 1

$$\begin{aligned} \max Z &= x_2 \\ -x_1 + x_2 &\leq \frac{1}{2} \\ x_1 + x_2 &\leq 3\frac{1}{2} \\ x_1, x_2 &\geq 0 \text{ et entiers} \end{aligned}$$



Arrondissement : exemple 2

$$\begin{aligned} \max Z &= x_1 + 5x_2 \\ x_1 + 10x_2 &\leq 20 \\ x_1 &\leq 2 \\ x_1, x_2 &\geq 0 \text{ et entiers} \end{aligned}$$



## Approche par énumération

- Un modèle en nombres entiers *borné* (par exemple, un modèle avec uniquement des variables 0-1) possède un nombre fini de solutions
- Pourquoi ne pas les énumérer toutes?
- Déjà, pour  $n=20$  variables 0-1, il y a plus d'un million de solutions possibles!
- Pour  $n=30$ , c'est plus d'un milliard!...
- Peut-être peut-on combiner cette idée d'énumérer les solutions avec l'idée de résoudre la relaxation PL pour éliminer certaines de ces solutions?

## Construction de l'arbre des solutions

- Un algorithme simple pour énumérer toutes les solutions d'un modèle 0-1 consiste à :
  - Choisir une variable  $x$
  - Générer les deux alternatives  $x=0$  et  $x=1$  (on dit que  $x$  est *fixée*) : chaque alternative correspond à un sommet de l'arbre des solutions
  - Recommencer à partir d'un sommet pour lequel certaines variables ne sont pas encore fixées
- *Racine* de l'arbre : aucune variable n'est encore fixée
- *Feuilles* de l'arbre : toutes les variables ont été fixées
- Nombre de feuilles =  $2^n$  (pour  $n$  variables 0-1)

## Algorithme de branch-and-bound

- Approche *diviser-pour-régner* :
  - Décomposition du problème en sous-problèmes plus simples
  - Puis combinaison de la résolution de ces sous-problèmes pour obtenir la solution du problème original
- Dans l'algorithme de *branch-and-bound (B&B)*, chaque sous-problème correspond à un sommet dans l'arbre des solutions
- On résout la relaxation PL de chaque sous-problème
- L'information tirée de la relaxation PL nous permettra (peut-être) d'éliminer toutes les solutions pouvant être obtenues à partir de ce sommet

## Algorithme de B&B : cas 0-1

- *Branchement* (ou séparation) :
  - Choisir un sommet dans l'arbre des solutions
  - Puis choisir une variable non encore fixée relativement à ce sommet
  - Générer les deux alternatives
- *Calcul de borne* (ou évaluation) : résoudre la relaxation PL en chaque sommet
- *Élagage* (ou élimination) : utiliser l'information tirée de la résolution de la relaxation PL pour éliminer toutes les solutions émanant du sommet courant
- B&B = *séparation et évaluation progressive*

## Algorithme de B&B : exemple

- Reprenons le problème *California Mfg*

$$\begin{array}{rcll}
 \max Z = & 9x_1 & +5x_2 & +6x_3 & +4x_4 & & \\
 & 6x_1 & +3x_2 & +5x_3 & +2x_4 & \leq & 10 \\
 & & & x_3 & +x_4 & \leq & 1 \\
 -x_1 & & & & & \leq & 0 \\
 & -x_2 & & & & \leq & 0 \\
 x_1, & x_2, & x_3, & x_4 & & & \text{binaire}
 \end{array}$$

- Relaxation PL : les variables peuvent prendre des valeurs fractionnaires entre 0 et 1
- Solution :  $(x_1, x_2, x_3, x_4) = (5/6, 1, 0, 1)$  et  $Z = 33/2$
- Branchons sur la variable  $x_1$

4. Programmation entiers

9

## Exemple : calcul de borne

- Sous-problème 1 :  $x_1 = 0$

$$\begin{array}{rcll}
 \max Z_1 = & 5x_2 & +6x_3 & +4x_4 & & \\
 & 3x_2 & +5x_3 & +2x_4 & \leq & 10 \\
 & & x_3 & +x_4 & \leq & 1 \\
 & & +x_3 & & \leq & 0 \\
 -x_2 & & & & & \leq & 0 \\
 x_2, & x_3, & x_4 & & & & \text{binaire}
 \end{array}$$

- Solution de la relaxation PL :

$$(x_1, x_2, x_3, x_4) = (0, 1, 0, 1) \text{ et } Z = 9$$

4. Programmation entiers

10

## Exemple : calcul de borne

- Sous-problème 2 :  $x_1 = 1$

$$\begin{array}{rcll}
 \max Z_2 = & 5x_2 & +6x_3 & +4x_4 & & +9 \\
 & 3x_2 & +5x_3 & +2x_4 & \leq & 4 \\
 & & x_3 & +x_4 & \leq & 1 \\
 & & +x_3 & & \leq & 1 \\
 -x_2 & & & & & \leq & 0 \\
 x_2, & x_3, & x_4 & & & & \text{binaire}
 \end{array}$$

- Solution de la relaxation PL :

$$(x_1, x_2, x_3, x_4) = (1, 4/5, 0, 4/5) \text{ et } Z = 16\frac{1}{5}$$

4. Programmation entiers

11

## Exemple : calcul de borne

- Sous-problème 1 :  $Z_1 \leq 9$
- Sous-problème 2 :  $Z_2 \leq 16 + 1/5$
- Notons que toutes les variables sont binaires et tous les paramètres dans l'objectif sont des valeurs entières
- Borne supérieure pour le sous-problème 2 : 16
- Pour le sous-problème 1, la solution obtenue est entière : c'est la *meilleure solution courante*
- On sait que la valeur optimale cherchée,  $Z$ , sera au moins  $Z^* = 9$  :  $Z \geq Z^*$

4. Programmation entiers

12

## Exemple : élagage

- Sous-problème 1 :
  - La solution optimale de la relaxation PL est entière
  - Il ne sert donc à rien de brancher sur les autres variables, puisque toutes les autres solutions entières (avec  $x_1 = 0$ ) sont nécessairement de valeur  $\leq 9$ !
  - On peut donc élaguer ce sommet (couper la branche!)
- Sous-problème 2 :
  - La solution optimale de la relaxation PL n'est pas entière
  - $Z^* = 9 \leq Z \leq 16$  : la branche ( $x_1 = 1$ ) peut encore contenir une solution optimale
  - Mais si on avait eu  $Z_2 \leq Z^*$ , on aurait pu conclure que la branche ne pouvait améliorer la meilleure solution courante

## Critères d'élagage

- Un sous-problème est élagué si une des trois conditions suivantes est satisfaite :
  - *Test 1* : Sa borne supérieure (valeur optimale de la relaxation PL) est  $\leq Z^*$  (valeur de la meilleure solution courante)
  - *Test 2* : Sa relaxation PL n'a pas de solution réalisable
  - *Test 3* : La solution optimale de sa relaxation PL est entière
- Lorsque le test 3 est vérifié :
  - On teste si la valeur optimale de la relaxation PL du sous-problème,  $Z_j$ , est supérieure à  $Z^*$
  - Si  $Z_j > Z^*$ , alors  $Z^* = Z_j$ , et on conserve la solution, qui devient la meilleure solution courante

## Algorithme de B&B : résumé

1. Initialisation :
  - a. Poser  $Z^* = -\infty$
  - b. Appliquer le calcul de borne et les critères d'élagage à la racine (aucune variable fixée)
2. Critère d'arrêt : s'il n'y a plus de sous-problèmes non élagués, arrêter
3. Branchement :
  - a. Parmi les sous-problèmes non encore élagués, choisir celui qui a été *créé le plus récemment* (s'il y a égalité, choisir celui de plus grande borne supérieure)
  - b. Appliquer le Test 1 : si le sous-problème est élagué, retourner en 2.
  - c. Brancher sur la prochaine variable non fixée

## Algorithme de B&B : résumé (suite)

4. Calcul de borne :
  - a. Résoudre la relaxation PL de chaque sous-problème
  - b. Arrondir la valeur optimale si tous les paramètres de l'objectif sont entiers
5. Élagage : élaguer un sous-problème si
  - a. La borne supérieure est  $\leq Z^*$
  - b. La relaxation PL n'a pas de solution réalisable
  - c. La solution optimale de la relaxation PL est entière : si la borne supérieure est  $> Z^*$ ,  $Z^*$  est mise à jour et la solution de la relaxation PL devient la meilleure solution courante
6. Retourner en 2.

## Règle de sélection

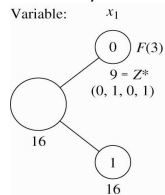
- Dans cette version, on propose comme *règle de sélection* de choisir le sous-problème le plus récemment créé
- Avantage : facilite la *réoptimisation* lors du calcul de borne, car peu de changements apportés par rapport au dernier sous-problème traité
- Désavantage : peut créer un grand nombre de sous-problèmes
- Autre option : *règle de la meilleure borne* (choisir le sous-problème ayant la plus grande borne supérieure)

## Règle de branchement

- Dans cette version, la règle de branchement consiste à choisir la prochaine variable non fixée
- Il est souvent plus intéressant de choisir une variable à *valeur fractionnaire*
- En branchant sur une telle variable, il est certain que les deux sous-problèmes créés mènent à des solutions différentes de la solution courante
- De nombreux critères existent pour choisir une telle variable de façon à orienter la recherche vers un élagage rapide

## Exemple (suite)

- Jusqu'à maintenant, voici l'arbre obtenu :



- F(3) indique que le sous-problème a été élagué (*fathomed*) en raison du Test 3

## Exemple (suite)

- Sélection : on choisit le sous-problème 2, le seul qui n'a pas encore été élagué
- On branche sur la prochaine variable, soit  $x_2$
- Deux nouveaux sous-problèmes sont créés :
  - Sous-problème 3 :  $x_1 = 1, x_2 = 0$
  - Sous-problème 4 :  $x_1 = 1, x_2 = 1$

## Exemple (suite)

- Sous-problème 3 :  $x_1 = 1, x_2 = 0$

$$\begin{array}{rcll} \max & Z_3 = & 6x_3 & + 4x_4 & + 9 \\ & & 5x_3 & + 2x_4 & \leq 4 \\ & & x_3 & + x_4 & \leq 1 \\ & & x_3 & & \leq 1 \\ & & & x_4 & \leq 0 \\ & & x_3, & x_4 & \text{binaire} \end{array}$$

- Solution de la relaxation PL :

$$(x_1, x_2, x_3, x_4) = (1, 0, 4/5, 0) \text{ et } Z = 13 \frac{4}{5} : Z_3 \leq 13$$

4. Programmation entiers

21

## Exemple (suite)

- Sous-problème 4 :  $x_1 = 1, x_2 = 1$

$$\begin{array}{rcll} \max & Z_4 = & 6x_3 & + 4x_4 & + 14 \\ & & 5x_3 & + 2x_4 & \leq 1 \\ & & x_3 & + x_4 & \leq 1 \\ & & x_3 & & \leq 1 \\ & & & x_4 & \leq 1 \\ & & x_3, & x_4 & \text{binaire} \end{array}$$

- Solution de la relaxation PL :

$$(x_1, x_2, x_3, x_4) = (1, 1, 0, 1/2) \text{ et } Z = 16 : Z_4 \leq 16$$

4. Programmation entiers

22

## Exemple (suite)

- Aucun des tests d'élagage ne s'applique sur ces sous-problèmes
- On doit donc choisir un des deux sous-problèmes pour effectuer un branchement, puisque ce sont ceux créés le plus récemment
- On choisit celui de plus grande borne supérieure, soit le sous-problème 4
- On branche sur  $x_3$  et on génère deux nouveaux sous-problèmes

4. Programmation entiers

23

## Exemple (suite)

- Sous-problème 5 :  $x_1 = 1, x_2 = 1, x_3 = 0$

$$\begin{array}{rcll} \max & Z_5 = & 4x_4 & & + 14 \\ & & 2x_4 & \leq & 1 \\ & & x_4 & \leq & 1 \\ & & x_4 & \leq & 1 \\ & & x_4 & & \text{binaire} \end{array}$$

- Solution de la relaxation PL :

$$(x_1, x_2, x_3, x_4) = (1, 1, 0, 1/2) \text{ et } Z = 16 : Z_5 \leq 16$$

4. Programmation entiers

24

## Exemple (suite)

- Sous-problème 6 :  $x_1 = 1, x_2 = 1, x_3 = 1$

$$\begin{aligned} \max \quad Z_6 &= 4x_1 && + 20 \\ & 2x_1 &\leq & -4 \\ & x_1 &\leq & 0 \\ & x_1 &\leq & 1 \\ & x_1 && \text{binaire} \end{aligned}$$

- La relaxation PL n'a pas de solution réalisable : ce sous-problème est élagué

## Exemple (suite)

- Le sous-problème 5 ne peut pas être élagué
- Il est créé le plus récemment parmi les sous-problèmes non élagués (3 et 5), on le choisit pour effectuer un branchement
- On branche sur  $x_4$  et on génère :
  - Sous-problème 7 :  $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0$
  - Sous-problème 8 :  $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 1$
- Toutes les variables sont fixées : on peut résoudre directement ces sous-problèmes

## Exemple (suite)

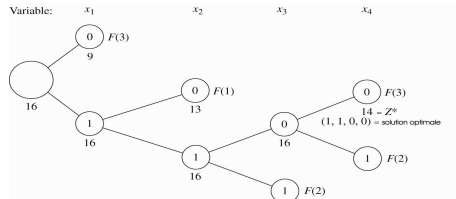
- Sous-problème 7 :  $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 0$ 
  - Solution :  $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$  et  $Z_7 = 14$
  - La solution est entière : le sous-problème est élagué par le Test 3
  - Puisque  $Z_7 > Z^*$ ,  $Z^* = 14$  et la solution du sous-problème devient la meilleure solution courante
- Sous-problème 8 :  $x_1 = 1, x_2 = 1, x_3 = 0, x_4 = 1$ 
  - Cette solution n'est pas réalisable, car la première contrainte ( $2x_4 \leq 1$ ) est violée
  - Le sous-problème est élagué par le Test 2

## Exemple (suite)

- Le sous-problème 3 est le seul non encore élagué
- On applique le Test 1 :  $Z_3 = 13 \leq 14 = Z^*$
- Le sous-problème est donc élagué
- Il n'y a plus de sous-problèmes non élagués : on arrête
- La solution optimale est :  
 $(x_1, x_2, x_3, x_4) = (1, 1, 0, 0)$  et  $Z = Z^* = 14$

## Exemple (suite et fin)

- Voici l'arbre obtenu suite à l'exécution de l'algorithme



- $F(j)$  : le sous-problème est élagué par le Test  $j$
- Voir autre exemple dans [IOR Tutor](#)

4. Programmation entiers

29

## Algorithme de B&B : cas général

- Cas général d'un modèle de programmation (mixte) en nombres entiers : variables entières générales et variables continues
- On modifie le branchement ainsi :
  - On choisit la première variable entière à *valeur non entière*
  - Soit  $x_j$  cette variable de valeur  $x_j^*$
  - Soit  $\lfloor x_j^* \rfloor =$  plus grand entier  $\leq x_j^*$
  - On génère deux nouveaux sous-problèmes :
 
$$x_j \leq \lfloor x_j^* \rfloor \text{ et } x_j \geq \lfloor x_j^* \rfloor + 1$$

4. Programmation entiers

30

## Algorithme de B&B : cas général

- Initialisation :
  - Poser  $Z^* = -\infty$
  - Appliquer le calcul de borne et les critères d'élagage à la racine (aucune variable fixée)
- Critère d'arrêt : s'il n'y a plus de sous-problèmes non élagués, arrêter
- Branchement :
  - Parmi les sous-problèmes non encore élagués, choisir celui qui a été *créé le plus récemment* (s'il y a égalité, choisir celui de plus grande borne supérieure)
  - Appliquer le Test 1 : si le sous-problème est élagué, retourner en 2.
  - Brancher sur la prochaine variable entière à valeur non entière dans la relaxation PL

4. Programmation entiers

31

## Algorithme de B&B (suite)

- Calcul de borne : résoudre la relaxation PL de chaque sous-problème
- Élagage : élaguer un sous-problème si
  - La borne supérieure est  $\leq Z^*$
  - La relaxation PL n'a pas de solution réalisable
  - Dans la solution optimale de la relaxation PL, toutes les variables entières sont à valeurs entières : si la borne supérieure est  $> Z^*$ ,  $Z^*$  est mise à jour et la solution de la relaxation PL devient la meilleure solution courante
- Retourner en 2.

4. Programmation entiers

32

## Algorithme de B&B : exemple

$$\begin{array}{rcll}
 \max Z = & 4x_1 & -2x_2 & +7x_3 & -x_4 & & & \\
 & x_1 & & +5x_2 & & & & \leq 10 \\
 & x_1 & +x_2 & -x_3 & & & & \leq 1 \\
 & 6x_1 & -5x_2 & & & & & \leq 0 \\
 & -x_1 & & +2x_2 & -2x_3 & & & \leq 3 \\
 & x_1 & & x_2 & & x_3 & & \geq 0 \\
 & x_1 & & x_2 & & x_3 & & \text{entier}
 \end{array}$$

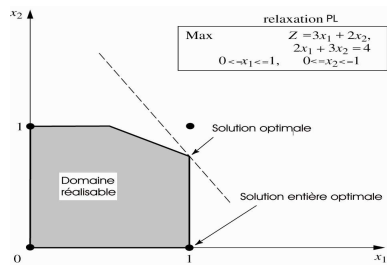
- Faire cet exemple avec [IOR Tutorial](#)

## Méthode de coupes

- Idée : ajouter des contraintes redondantes pour le modèle en nombres entiers, mais non pour la relaxation PL
- Exemple :  $\max Z = 3x_1 + 2x_2$   
 $2x_1 + 3x_2 \leq 4$   
 $x_1, x_2$  binaire
- Les solutions réalisables sont (0,0), (1,0) et (0,1)
- Une contrainte redondante est :  $x_1 + x_2 \leq 1$

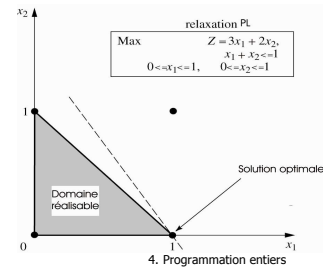
## Méthode de coupes : exemple

- Domaine réalisable de la relaxation PL :



## Méthode de coupes : exemple

- Suite à l'ajout de la contrainte redondante, le problème est résolu à la racine!





## Méthode de coupes

- Il y a plusieurs algorithmes permettant de générer de telles inégalités redondantes, appelées *coupes*
- Mais il est rare que leur ajout permet de résoudre le problème à la racine
- L'ajout de coupes permet toutefois de réduire le nombre de sous-problèmes traités par l'algorithme de B&B
- On peut même ajouter des coupes pour chaque sous-problème (pas seulement à la racine) : on obtient alors un algorithme de branch-and-cut



## Pour expérimenter avec B&B

- Pour des petits modèles (moins de 5 variables de *tous types* et moins de 5 contraintes fonctionnelles) : essayer le [IOR Tutorial](#)
- Pour des modèles plus gros, modéliser et résoudre avec *Excel Solver*
- Revoir le cas [California Mfg](#)
- Pour des modèles encore plus gros, essayer Lindo/Lingo et CPLEX/MPL (CD)